C4 - Natural Language Processing

C-DAT-500

Topic modeling

from 52 books of English literature

2.0

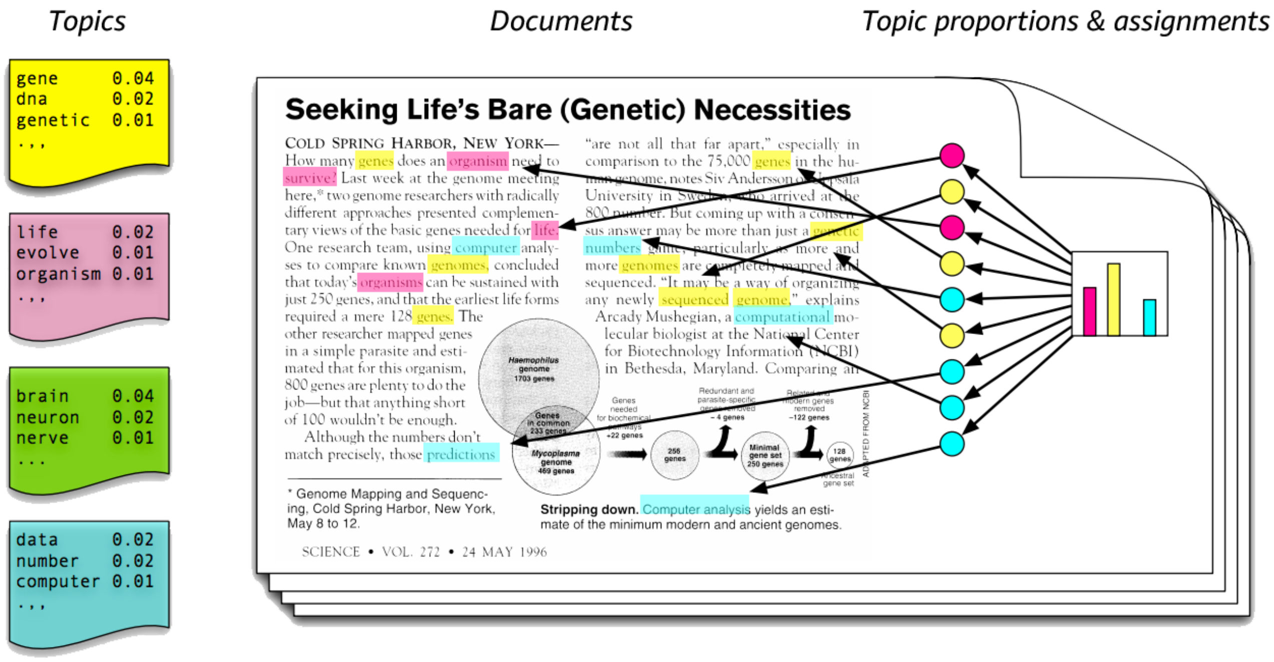
Topic modeling

delivery method: nlp-alice on Github
language: python

The idea behind Natural Language Processing, **NLP**, is the same as doing data analysis on tabular data **except its text**, not numbers.

The goal of this project is **to find the subject** of each text from the data. This is called **Topic Modeling**.

It's the essence of many application such as **Search Engine** or **Spam filters** for example.



Project Gutenberg Dataset

Project Gutenberg is the effort of Michael S. Hart to digitize and archive cultural works, as well as to encourage the creation and distribution of eBooks, **which he invented in 1971**.

The project reached **62 108 books**, most in the public domain. We'll use a subset of 52 books for this project.

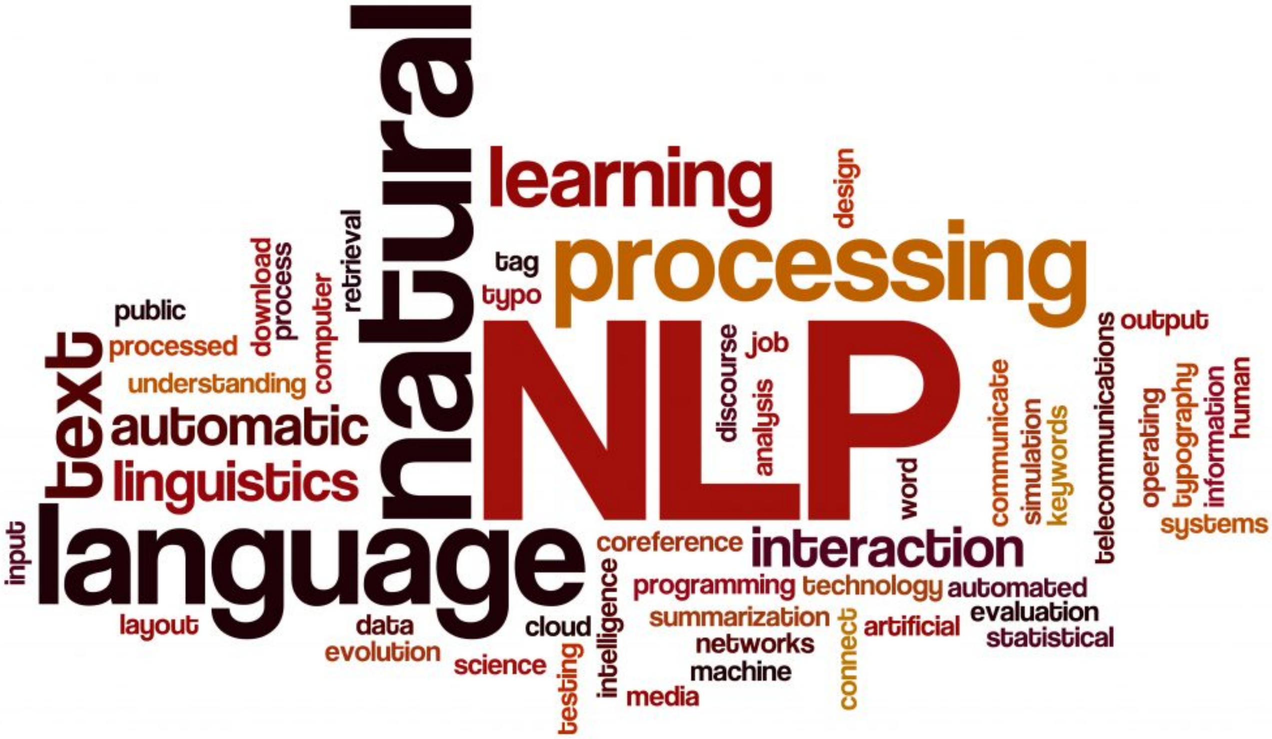
Part one : Words cloud

For each book, generate the **bag of words** and its associated **word cloud.**

A bag of words is a pandas dataframe with the **frequency of each word** for every word that occurs in the document.

The word order from the original text is non-relevant. The only thing kept are the unique words and their frequency.

As a preprocessing step, the grammar, punctuation, tenses, conjugations and prepositions, **must be stipped form the original text.**



only NLTK library is allowed for the preprocessing

Part two : Topic Modeling

You must use **sk-learn** library from now on.

TF-IDF

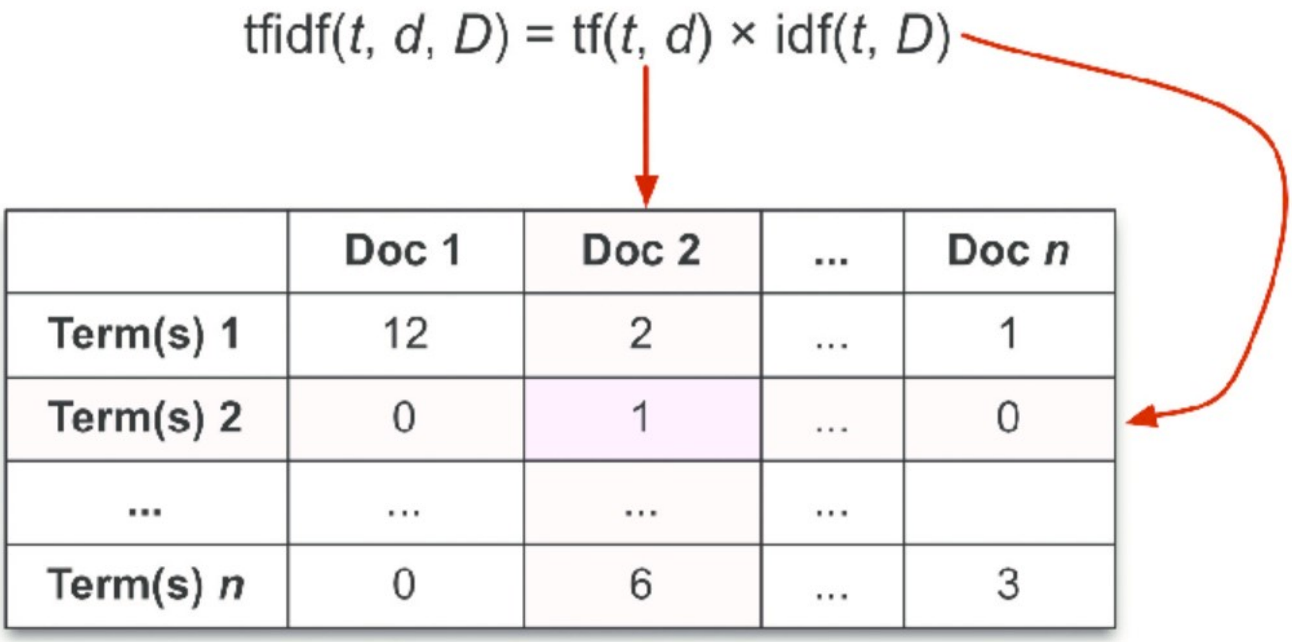
Now you must **find the subject** of each text using the data.

This part builds heavily on the concept of **vector representation of a document** based on the relative frequency of individual words **w.r.t the whole corpus** (all 52 books).

If we have 10,000 unique words across all the documents and 500 documents, we can create a 500 x 10,000 matrix, where :
- each collum is a vector representing a document,
- each row is the unique words,
- each element of the matrix represents the term frequency the word in a particular document.

We'll name it document_term_matrix.

Use **TfidfVectorizer** method from sk-learn to create this matrix.



$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \times \mathrm{idf}(t, D)$$

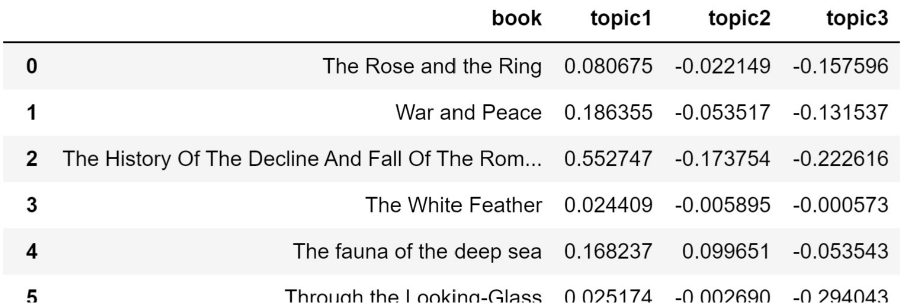
	Doc 1	Doc 2	...	Doc *n*
Term(s) 1	12	2	...	1
Term(s) 2	0	1	...	0
...	...	...	...	
Term(s) *n*	0	6	...	3

Latent Semantic Analysis (L.S.A)

As a next step we are going to transform those vectors into lower-dimension representation **in order to identify topics.**

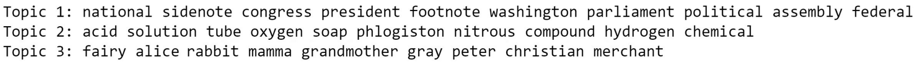
Perform LSA on your document_term_matrix.

Use **TruncatedSVD** (the name of LSA algorithm in sk-learn) method, using n_components = 3 to find your topics.



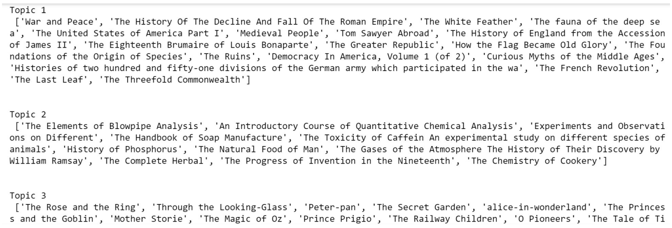
	book	topic1	topic2	topic3
0	The Rose and the Ring	0.080675	-0.022149	-0.157596
1	War and Peace	0.186355	-0.053517	-0.131537
2	The History Of The Decline And Fall Of The Rom...	0.552747	-0.173754	-0.222616
3	The White Feather	0.024409	-0.005895	-0.000573
4	The fauna of the deep sea	0.168237	0.099651	-0.053543
5	Through the Looking-Glass	0.025174	-0.002690	-0.294043

For each topic, display its 10th most frequent words.



```
Topic 1: national sidenote congress president footnote washington parliament political assembly federal
Topic 2: acid solution tube oxygen soap phlogiston nitrous compound hydrogen chemical
Topic 3: fairy alice rabbit mamma grandmother gray peter christian merchant
```

For each book, link its associated topic by using the max value from the lsa_topic_matrix in order to display all books from each topic.

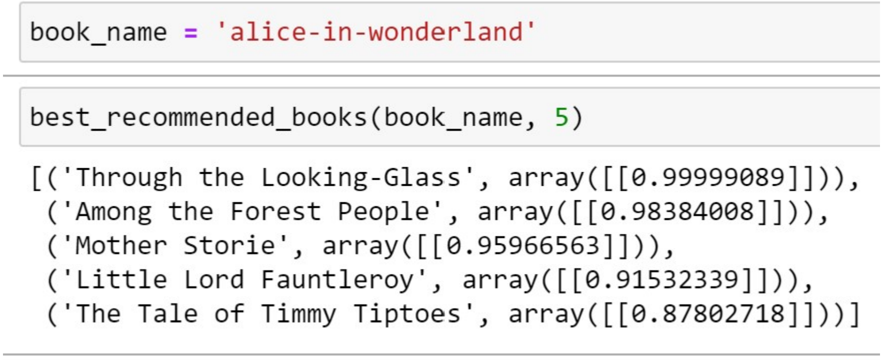


```
Topic 1
 ['War and Peace', 'The History Of The Decline And Fall Of The Roman Empire', 'The White Feather', 'The fauna of the deep se
a', 'The United States of America Part I', 'Medieval People', 'Tom Sawyer Abroad', 'The History of England from the Accession
of James II', 'The Eighteenth Brumaire of Louis Bonaparte', 'The Greater Republic', 'How the Flag Became Old Glory', 'The Fou
ndations of the Origin of Species', 'The Ruins', 'Democracy In America, Volume 1 (of 2)', 'Curious Myths of the Middle Ages',
'Histories of two hundred and fifty-one divisions of the German army which participated in the wa', 'The French Revolution',
'The Last Leaf', 'The Threefold Commonwealth']


Topic 2
 ['The Elements of Blowpipe Analysis', 'An Introductory Course of Quantitative Chemical Analysis', 'Experiments and Observati
ons on Different', 'The Handbook of Soap Manufacture', 'The Toxicity of Caffein An experimental study on different species of
animals', 'History of Phosphorus', 'The Natural Food of Man', 'The Gases of the Atmosphere The History of Their Discovery by
William Ramsay', 'The Complete Herbal', 'The Progress of Invention in the Nineteenth', 'The Chemistry of Cookery']


Topic 3
 ['The Rose and the Ring', 'Through the Looking-Glass', 'Peter-pan', 'The Secret Garden', 'alice-in-wonderland', 'The Princes
s and the Goblin', 'Mother Storie', 'The Magic of Oz', 'Prince Prigio', 'The Railway Children', 'O Pioneers', 'The Tale of Ti
```

PART 3 : DOCUMENT SIMILARITY

One real cool application of having vectors representation of books is that you can create a **recommender system** by using the cosine distance of two vectors (of two books).

Create a function **best_recommended_books** that suggest to the buyer of a book the 5 most related books to buy.



```
book_name = 'alice-in-wonderland'
```

```
best_recommended_books(book_name, 5)
```

```
[('Through the Looking-Glass', array([[0.99999089]])),
 ('Among the Forest People', array([[0.98384008]])),
 ('Mother Storie', array([[0.95966563]])),
 ('Little Lord Fauntleroy', array([[0.91532339]])),
 ('The Tale of Timmy Tiptoes', array([[0.87802718]]))]
```